Eithne: A framework for benchmarking micro-core accelerators

Eithne: a framework for benchmarking micro-core accelerators

Maurice Jamieson
Nick Brown
EPCC at the University of Edinburgh
THE UNIVERSITY of EDINBURGH

What are micro-cores?

- Micro-cores have small instructions sets and tiny amounts (c.32KB) of on-chip memory, resulting in very low power consumption
- The low power consumption and simplicity of core mean that a large number can be placed on a chip
- The HPC community is more and more interested in micro-core architectures as they provide massive parallelism on -chip. For example the RISC-V based European Processor Initiative (EPI), and a combination of micro-cores with normal technology for "posits"
- There are lots of hard-processor examples including the Adapteva Epiphany and the PEZY-SC2 (Shoubu system B)
- There is also an increasing number of soft-core examples, such as the GRVI Phalanx

Which one to choose?

There are a number of factors to consider for the selection of a soft-core for use in a micro-core accelerator:

- Performance
- Power consumption
- Chip area - complex instruction sets require large decoding logic
- Scalability - maximum clock frequency can be limited by the complexity of core design
- Code density - on-chip RAM limited

Lots and lots of choice...

Framework overview

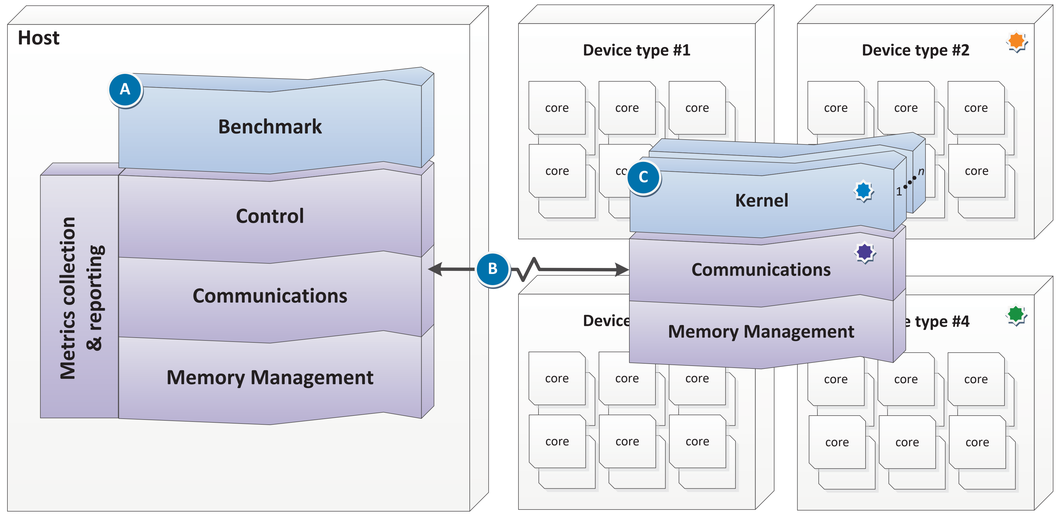
The framework provides a set API functions that enable a 'plug-in' architecture to support multiple benchmarks and devices

The framework is modular and simple to extend:

- New devices
- New comms link

No modifications required to existing benchmark / kernel codes

The definition of "device" is flexible: most often this is a micro-core accelerator but it could be a thread running on the host



Kernels are device agnostic: the same kernel can be run on all supported devices without modification

A Host benchmark code

```
buffer = EITHNE_ALLOC_MEM(sizeof(float)*N*LDA);

EITHNE_INIT_HOST(vars, HOST_ID, buffer + EITHNE_DATA_OFFSET, buffer);

EITHNE_INIT_CORES(16);
EITHNE_START_CORES(16);
```

- The host benchmark application includes four API calls to allocate memory, initialise and start the device core(s)

B Communications / control code

```
EITHNE_SEND(vars, TARGET_ID, A);

t1 = cpu_time ( );

EITHNE_EXECUTE(TARGET_ID, SGEFA);

t2 = cpu_time ( );

EITHNE_RECV(vars, TARGET_ID, A);
EITHNE_RECV(vars, TARGET_ID, IPVT);
EITHNE_RECV(vars, TARGET_ID, RESULT);
```

- The transfer of shared variable data and launching of kernel codes is controlled by API calls within the host application
- Eithne supports multi-core architectures e.g. the TARGET_ID core ID in the EITHNE_RECV API call

C Device kernel code

```
void kernel_init(EithneTargetId id, EithneSharedMem buffer) {
  EithneKernel kernels[2];

  kernels[SGEFA] = sgefa;
  kernels[SGESL] = sgesl;

  EITHNE_INIT_DEVICE(vars, id, buffer + EITHNE_DATA_OFFSET, buffer, kernels);

  EITHNE_REGISTER_ARRAY(vars, A, EITHNE_FLOAT_ARRAY, a, N*LDA);
  EITHNE_REGISTER_ARRAY(vars, B, EITHNE_FLOAT_ARRAY, b, N);
  EITHNE_REGISTER_ARRAY(vars, IPVT, EITHNE_INTEGER_ARRAY, ipvt, N);
  EITHNE_REGISTER_SCALAR(vars, JOB, EITHNE_INTEGER, job);
  EITHNE_REGISTER_SCALAR(vars, RESULT, EITHNE_INTEGER, info);

  EITHNE_START_LISTENER;
}
```

- The device kernel code just includes the API calls to initialise the device, register the shared variables and start the communications listener (no other code changes are required)

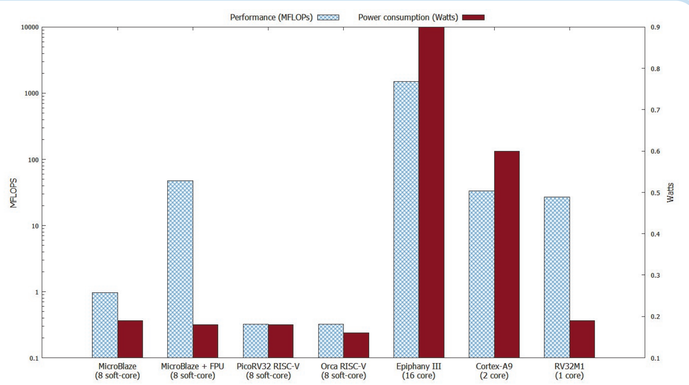
This is the only additional code that needs to be added to device kernels

Objectives

- Provide a framework to support benchmarking of multiple hard and soft micro -core accelerators from a single codebase
- Measurement:
  - FLOPS
  - Power consumption (Watts)
  - Code size
- Support multiple benchmarks

Sample Results

- For expediency, we highlight a small sample of results here:
  - Performance and power consumption for a single core of a subset of the supported devices
- Eithne supports multi-core devices and metrics can be created for a group / cluster of cores
- The results highlight the benefits of a framework that supports a number of different micro-cores and communication links



Framework extensibility

- Eithne is a stack of functionality:
  - **Benchmarks**
  - **Devices**
  - **Kernels**
  - **Communications (I/O)**
  - **Metrics**
- New functionality can be added at each level without impacting others
- Kernels are typically written in C and can be tailored to a specific device using the provided APIs, if the user wishes

Currently supported devices

- **RISC-V**
  - PicoRV32 (soft-core)
  - VectorBlox Orca (soft-core)
  - RI5CY (NXP NV32M1)
- **Xilinx MicroBlaze (soft-core)**
- **ARM**
  - Cortex-M1 (soft-core)
  - Cortex-A9
- **Adapteva Epiphany III**
- **Intel x86-64**

Further work

- Implement additional benchmarks
- Benchmark additional RISC-V soft-cores e.g. RI5CY, SweRV
- Kernels implemented using OpenMP
- MPI-based communications

GitLab

Available on GitLab:

https://gitlab.com/mjamieson/eithne